JOÃO MANUEL CONCEIÇÃO VIEIRA 30333

VERSAO A

PARTE I

1.
A imagem ilustra a interação entre o ambiente do lado do cliente (client-side) e o ambiente do lado do servidor (server-side) na web.

No lado esquerdo, representando o ambiente do lado do cliente, temos um computador com um navegador da web (como Firefox, Chrome, Internet Explorer). Este lado inclui tecnologias como JavaScript, HTML e CSS, que são responsáveis pela aparência e comportamento da interface do utilizador no navegador.

No lado direito, representando o ambiente do lado do servidor, temos um servidor que processa pedidos e responde a eles. Este lado inclui tecnologias como PHP, Java e JSP, Node.js, ASP, Python e Ruby on Rails. Estas tecnologias são usadas para processar a lógica de negócios, aceder a bases de dados e gerar o conteúdo que será enviado de volta ao cliente.

O diagrama mostra uma comunicação bidirecional entre o cliente e o servidor via HTTP, com o cliente a enviar um pedido (request) e o servidor a responder (response). Isto ilustra a arquitetura básica de aplicações web, onde o cliente solicita informações ou serviços e o servidor processa esses pedidos e envia as respostas adequadas de volta ao cliente.

2.

Protocolo para Exames de Programação Web no IPVC

Objetivo

O protocolo para exames de programação web no IPVC visa estabelecer normas e procedimentos claros e uniformes para a realização de exames pelos alunos, garantindo a qualidade, a justiça e a transparência do processo avaliativo.

Funcionalidades

O protocolo abrange os seguintes aspectos:

- **Inscrição em exames:** Define os prazos, procedimentos e documentação necessária para os alunos se inscreverem nos exames.

- **Local e data dos exames:** Informa os locais e datas dos exames, bem como os horários de entrada e saída dos alunos.
- **Ambiente de desenvolvimento:** Especifica o ambiente de desenvolvimento que os alunos utilizarão durante o exame, incluindo o sistema operacional, editor de código, IDE e compilador.
- **Materiais permitidos:** Define os materiais que os alunos podem levar para os exames, como lápis, caneta, borracha, calculadora e documentação relevante.
- **Comportamento durante o exame:** Estabelece as regras de conduta que os alunos devem seguir durante o exame, incluindo a proibição de comunicação, utilização de materiais não permitidos e plágio.
- **Avaliação dos exames:** Define os critérios de avaliação dos exames, bem como o processo de revisão e contestação das notas.
- **Reclamações e recursos:** Informa os procedimentos para os alunos apresentarem reclamações ou recursos contra as notas dos exames.

Importância do Protocolo

O protocolo para exames é importante por vários motivos:

- **Garantir a qualidade do processo avaliativo:** O protocolo garante que os exames sejam realizados de forma justa e transparente, com critérios de avaliação claros e consistentes.
- **Promover a justiça para os alunos:** O protocolo garante que todos os alunos tenham as mesmas oportunidades de sucesso nos exames, independentemente de fatores externos.
- **Evitar conflitos e mal-entendidos:** O protocolo ajuda a evitar conflitos e mal-entendidos entre alunos e docentes, pois define claramente as regras e procedimentos para a realização dos exames.
- **Promover a transparência:** O protocolo garante que o processo avaliativo seja transparente para todos os envolvidos, incluindo alunos, docentes e a comunidade académica.

Exemplo de Protocolo para Exames de Programação Web

1. Inscrição em Exames

- Os alunos devem se inscrever nos exames através do sistema online do IPVC, dentro do prazo estabelecido no calendário letivo.
- No momento da inscrição, os alunos devem apresentar o seu cartão de estudante e o comprovativo de pagamento da propina.
- Os alunos que não se inscreverem dentro do prazo terão de pagar uma taxa adicional.

2. Local e Data dos Exames

- Os locais e datas dos exames serão publicados no website do IPVC e no sistema online dos alunos.

- Os alunos devem chegar aos locais dos exames com antecedência de pelo menos 15 minutos.
- Os alunos que chegarem atrasados poderão não ser autorizados a realizar o exame.

3. Ambiente de Desenvolvimento

- Os alunos utilizarão o seguinte ambiente de desenvolvimento durante o exame:
    - Sistema operacional: Windows 10
    - Editor de código: Visual Studio Code
    - IDE: IntelliJ IDEA
    - Compilador: GCC

4. Materiais Permitidos

- Os alunos apenas poderão levar para os exames os seguintes materiais:
    - Lápis
    - Caneta
    - Borracha
    - Calculadora
    - Documentação relevante da disciplina (apresentações, exercícios, etc.)

5. Comportamento durante o Exame

- Os alunos devem manter um comportamento silencioso e ordeiro durante o exame.
- É proibida a comunicação entre os alunos durante o exame.
- A utilização de telemóveis ou outros dispositivos eletrónicos durante o exame é estritamente proibida.
- O plágio é considerado uma falta grave e pode resultar na anulação do exame.

6. Avaliação dos Exames

- Os critérios de avaliação dos exames serão definidos pelo docente responsável pela unidade curricular.
- Os critérios de avaliação podem incluir:
    - Correção do código
    - Eficiência do código
    - Qualidade da documentação
    - Originalidade da solução
    - Capacidade de explicar o código

7. Reclamações e Recursos

- Os alunos que discordarem da sua nota podem apresentar uma reclamação ao docente responsável pela unidade curricular, no prazo de 5 dias úteis após a publicação das notas.
- Se a reclamação não for resolvida ao nível do docente, os alunos podem apresentar um recurso ao Conselho Pedagógico.

PARTE II

1.
document.getElementById(id):

Singular (getElement): Retorna um único elemento com o id especificado, pois cada valor de id deve ser único dentro de um documento HTML. Assim, sempre há (ou deve haver) apenas um elemento com esse id.
EXEMPLO NO GITHUB

document.getElementsByTagName(tagName):

Plural (getElements): Retorna uma coleção (HTMLCollection) de todos os elementos com o nome da tag especificado. Como pode haver múltiplos elementos com a mesma tag em um documento, este método retorna um conjunto de elementos.
EXEMPLO NO GITHUB

document.getElementsByClassName(className):

Plural (getElements): Retorna uma coleção (HTMLCollection) de todos os elementos que possuem a classe especificada. Como muitos elementos podem compartilhar a mesma classe, este método retorna um conjunto de elementos.
EXEMPLO NO GITHUB

document.getElementById(id): Retorna um único elemento (singular) com o id especificado.

document.getElementsByTagName(tagName): Retorna uma coleção de todos os elementos com o nome da tag especificado.

document.getElementsByClassName(className): Retorna uma coleção de todos os elementos com a classe especificada.

A diferença no uso de "Element" (singular) e "Elements" (plural) é que getElementById sempre retorna um único elemento, enquanto getElementsByTagName e getElementsByClassName retornam coleções de elementos.

2.
FEITO NO GITHUB

A escolha entre JSON e XML depende do contexto e das necessidades específicas do projeto. JSON é preferido para aplicações web modernas devido à sua simplicidade e desempenho, enquanto XML é útil para documentos complexos que requerem validação rigorosa.

PARTE III

1.

1º- src: "https://www.ipvc.pt/wp-content/uploads/2020/11/logo_ipvc_svg.svg"

Este é um URL absoluto, que inclui o esquema (https), o domínio (www.ipvc.pt), e o caminho completo para a imagem no servidor. Isso significa que, independentemente de onde o documento HTML esteja localizado, o navegador irá procurar a imagem a partir do endereço completo fornecido.

alt: "logo da IPVC"

Este atributo fornece um texto alternativo para a imagem, útil para acessibilidade e para o caso da imagem não ser carregada.

2º - src: "/images/logo_ipvc_svg.svg"

Este é um URL relativo, que indica o caminho para a imagem relativo à raiz do servidor. O navegador irá procurar a imagem no diretório /images/ no servidor onde o documento HTML está localizado. A localização exata depende da estrutura do servidor web em que o HTML está sendo servido.

alt: "logo da esm"

Este atributo fornece um texto alternativo diferente, "logo da esm".

Diferenças Específicas:

Caminho da Imagem (src):O primeiro src usa um URL absoluto, fornecendo um caminho completo para a imagem, o que garante que a imagem será carregada de um local específico na web.

O segundo src usa um URL relativo, que depende da localização do servidor e da estrutura do diretório onde o documento HTML está localizado.

Texto Alternativo (alt):

O primeiro alt é "logo da IPVC", indicando que a imagem é o logo do Instituto Politécnico de Viana do Castelo.

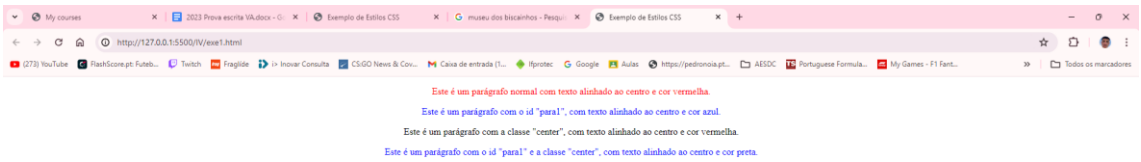
O segundo alt é "logo da esm", que provavelmente se refere ao logo de outra entidade ou departamento dentro da mesma instituição.

2.

O atributo lang no elemento <html> serve para declarar o idioma principal do conteúdo de um documento HTML. No exemplo <html lang="pt">, o valor "pt" indica que o conteúdo da página está em português.

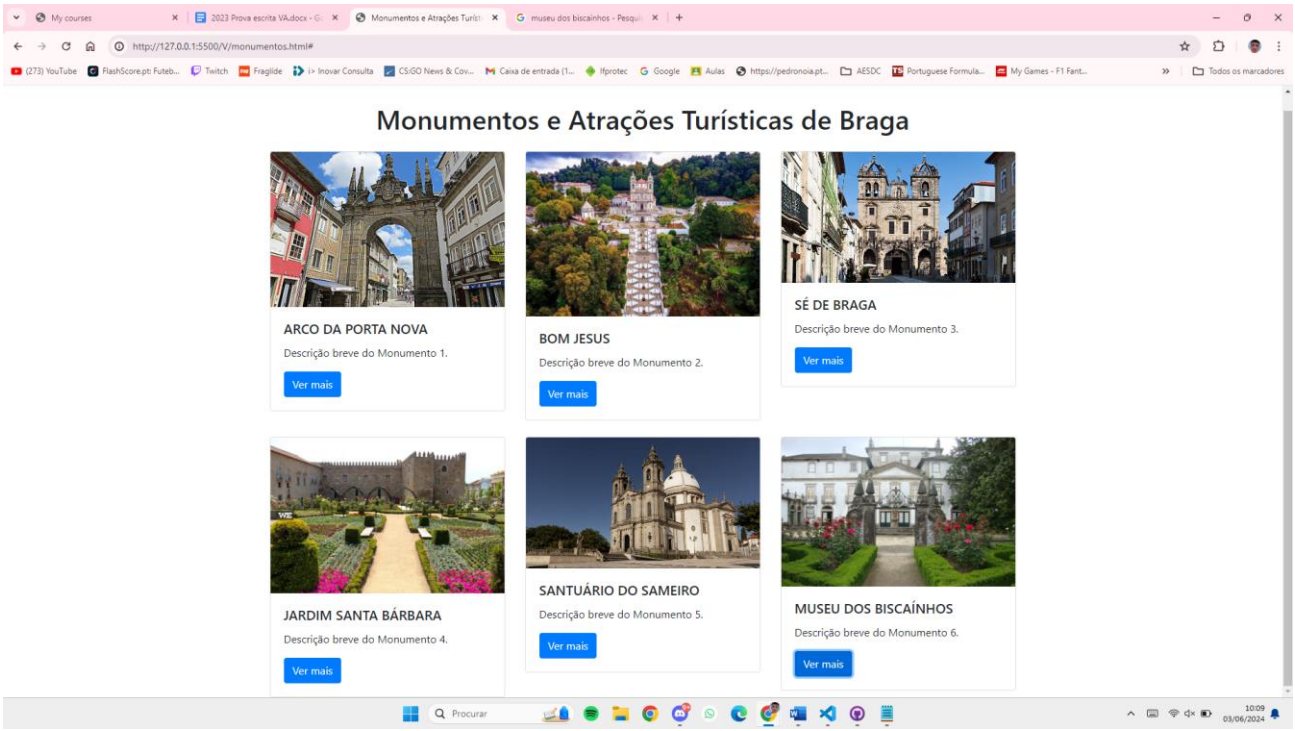
PARTE IV

CÓDIGO NO GITHUB



PARTE V

CÓDIGO NO GITHUB



PARTE VI

1.1- FEITO NO GITHUB

1.2-

const productsRouter = require('express').Router();

Esta linha importa o módulo express e cria uma nova instância do roteador (router). O roteador é utilizado para definir e agrupar rotas relacionadas ao recurso "produtos".

const controller = require('../controllers/products');

Aqui, o arquivo ../controllers/products é importado e armazenado na constante controller. Este arquivo é esperado que contenha os métodos do controlador que lidarão com a lógica de negócios das rotas (ex.: obter, criar, atualizar e excluir produtos).

const authMiddleware = require('../middlewares/auth/auth');

Esta linha importa o middleware de autenticação localizado em ../middlewares/auth/auth. O middleware é uma função que será executada antes de atingir os métodos do controlador, verificando se o usuário está autenticado.

productsRouter.get('/', authMiddleware, controller.getAll);

Define uma rota GET para o caminho raiz (/) da rota products. Quando esta rota é acessada, o middleware de autenticação é executado primeiro (authMiddleware). Se a autenticação for bem-sucedida, o método getAll do controlador (controller.getAll) é chamado para obter todos os produtos.

productsRouter.get('/:id', authMiddleware, controller.getById);

Define uma rota GET para o caminho / seguido de um parâmetro (:id). Esta rota é usada para obter um produto específico pelo seu id. Novamente, o middleware de autenticação é executado primeiro, e, se autorizado, o método getById do controlador é chamado.

productsRouter.post('/', authMiddleware, controller.create);

Define uma rota POST para o caminho raiz (/) da rota products. Esta rota é usada para criar um novo produto. O middleware de autenticação verifica a permissão antes de chamar o método

productsRouter.put('/:id', authMiddleware, controller.update);

Define uma rota PUT para o caminho / seguido de um parâmetro (:id). Esta rota é usada para atualizar um produto específico pelo seu id. O middleware de autenticação verifica se o usuário está autorizado antes de chamar o método update do controlador.

productsRouter.delete('/:id', authMiddleware, controller.delete);

Define uma rota DELETE para o caminho / seguido de um parâmetro (:id). Esta rota é usada para excluir um produto específico pelo seu id. O middleware de autenticação verifica se o usuário tem permissão para excluir antes de chamar o método delete do controlador.

module.exports = productsRouter;

Esta linha exporta o roteador configurado (productsRouter) para que ele possa ser usado em outras partes da aplicação, geralmente sendo importado no arquivo principal do servidor (ex.: app.js ou server.js) para integrar as rotas na aplicação principal.

Resumindo, este arquivo configura e exporta um conjunto de rotas protegidas por autenticação para manipular operações **CRUD (Create, Read, Update, Delete)** em produtos, delegando a lógica de negócios para métodos específicos em um controlador.

1.3 **FEITO NO GITHUB**